

Attorney Docket No. 2860-058; P2200/EJB

APPLICATION FOR UNITED STATES LETTERS PATENT FOR

**LATENCY-REDUCING BANDWIDTH-PRIORITIZATION  
FOR NETWORK SERVERS AND CLIENTS**

INVENTOR:  
Jakob Nielsen

PREPARED BY:  
David L. Stewart  
LOWE, PRICE, LEBLANC & BECKER  
99 Canal Center Plaza, Suite 300  
Alexandria, VA 22314  
Telephone: (703) 684-1111  
Facsimile: (703) 684-1124

2860-058

P2200/EJB

**LATENCY-REDUCING BANDWIDTH-PRIORITIZATION  
FOR NETWORK SERVERS AND CLIENTS**

BACKGROUND OF THE INVENTION

Field of the Invention

The invention relates to communication systems and more particularly to client server communication systems in which communications resources provided to a retrieval request or to a connection to a particular server can be adjusted to reflect priorities.

Description of Related Art

Accessing the content of servers on a network, such as the World Wide Web (WWW), is often agonizingly slow. Response time may be so slow that users cancel their request for a page and select another page, perhaps that of a competitor. Getting enough text to the user quickly so that he may begin reading is important to prevent him from canceling his request. Reducing the initial latency from request to delivery of readable text will result in a more satisfying browsing experience for the user.

Data compression is known. Various data compression algorithms may be employed depending upon the type of data (text, image, video, audio) to be transmitted. Fortunately, many if not most network servers employ compression. Unfortunately, the latency problems associated with retrieving server pages continue to grow worse with increasing network usage.

#### SUMMARY OF THE INVENTION

The present system provides apparatus, systems, methods and computer program products which will allow improved average response time in client server technology. This is accomplished by prioritizing the content of network pages and allocating server or client bandwidth accordingly. The prioritization scheme is driven by the need to render a viewable page as quickly as possible so that the user may begin reading it. In the prior art, all files that make up the final appearance of a WWW page (text, style sheets, graphics, audio etc.) are sent with equal priority. By using a prioritization scheme, it is possible to dramatically reduce the latency perceived by the user which, in turn, increases the perceived usability of the web site.

Two mechanisms facilitate this. First, the order in which information is displayed to the user "hides" some of the latency thus creating the perception of faster

response time. Second, by monitoring individual connections it is possible to determine which connections are not currently utilizing the bandwidth initially made available to them. When this occurs, the unused  
5 bandwidth is allocated to all other existing connections which might be able to use it.

The same concepts can be applied to the client-side (the user) in one form by allowing the client to indicate to the server the speed with which the client will accept  
10 information. In another form, a client that has multiple browsers connected to respective multiple network servers can prioritize each connection to insure, for example, that the current browser window's connection is allocated more bandwidth than a minimized browser window's  
15 connection. This is not possible using the prior art since all connections are given the same priority.

The invention is directed to computer apparatus, such as that found at a server, for allocating communications bandwidth to a plurality of user  
20 connections. The apparatus includes a processor configured to allocate communications bandwidth to said user connections based on at least one set of priorities.

The invention is also directed to computer apparatus, such as that running a client process, for  
25 allocating communications bandwidth to a plurality of

server connections based on at least one set of priorities.

The invention is also directed to a communications system operating on a network and having at least one server and at least one computer running a client process in which said at least one server or said at least one computer allocates bandwidth to a plurality of network connections based on at least one set of priorities.

The invention is also directed to methods for operating servers and clients to allocate bandwidth to connections based on at least one set of priorities.

The invention also relates to computer program products useful for carrying out the methods and for implementing the apparatus described.

The foregoing and other features, aspects and advantages of the present invention will become more apparent from the following detailed description of the present invention when taken in conjunction with the accompanying drawings.

#### BRIEF DESCRIPTION OF THE DRAWINGS

The object, features and advantages of the system of the present invention will be apparent from the following descriptions, in which:

**Figure 1A** illustrates a computer of a type suitable for carrying out the invention as either a client or a server.

**Figure 1B** illustrates a block diagram of the computer of **Figure 1A**.

**Figure 1C** illustrates an exemplary memory medium containing on or more programs usable with the computer of **Figure 1A**.

**Figure 2A** is an illustration of one exemplary form of implementing the invention using a network such as an intranet.

**Figure 2B** is an illustration of another exemplary form of implementing the invention using a typical internet arrangement.

**Figure 3** is an illustration of bandwidth allocation from a network server to several clients.

**Figure 4** is a database schema organized as an exemplary way for storing file-type priorities.

**Figure 5** is a database schema of an exemplary way for storing current connection information, namely in an Allocated Utilization Table (AUT).

**Figure 6** is a flowchart of a process to initiate dynamic bandwidth allocation by the server.

**Figure 7A** is a flowchart of a process for dynamic bandwidth allocation by the server.

**Figure 7B** is a illustration depicting the result of one iteration of the dynamic bandwidth allocation procedure of **Figure 7A**.

**Figure 8A** is a database schema organized as an exemplary way for storing client browser-status priorities.

**Figure 8B** is a flowchart of a process for client-side control of bandwidth allocation.

**Figure 9** is a flowchart of a process for changing the priority of a connection after a fixed amount of data has been transmitted.

**Figure 10A** is a database schema organized as an exemplary way for storing a value representing the respective amount of data specific files need to have transmitted at high priority.

**Figure 10B** is a flowchart of a process for changing the priority of a connection after the variable amount of data indicated in **Figure 10A** has been transmitted.

**Figure 11A** is a database schema organized as an exemplary way for storing a list of customer passwords and the corresponding priority multiplier.

**Figure 11B** is a flowchart of a process for increasing the priority of a connection using the information contained within **Figure 11A**.

**Figure 12A** is a database schema organized as an exemplary way for storing a list of documents and their associated priority multipliers.

**Figure 12B** is a flowchart of a process for increasing the priority of a connection using the information contained within **Figure 12A**.

#### NOTATIONS AND NOMENCLATURE

The detailed descriptions which follow may be presented in terms of program procedures executed on a computer or network of computers. These procedural descriptions and representations are the means used by those skilled in the art to most effectively convey the substance of their work to others skilled in the art.

A procedure is here, and generally, conceived to be a self-consistent sequence of steps leading to a desired result. These steps are those requiring physical manipulations of physical quantities. Usually, though not necessarily, these quantities take the form of electrical or magnetic signals capable of being stored, transferred, combined, compared, and otherwise manipulated. It proves convenient at times, principally for reasons of common usage, to refer to these signals as bits, values, elements, symbols, characters, terms, numbers, or the like. It should be noted, however, that all of these and similar terms are to be associated with



the appropriate physical quantities and are merely convenient labels applied to these quantities.

Further, the manipulations performed are often referred to in terms, such as adding or comparing, which are commonly associated with mental operations performed by a human operator. No such capability of a human operator is necessary, or desirable in most cases, in any of the operations described herein which form part of the present invention; the operations are machine operations.

Useful machines for performing the operation of the present invention include general purpose digital computers or similar devices.

The present invention also relates to apparatus for performing these operations. This apparatus may be specially constructed for the required purpose or it may comprise a general purpose computer as selectively activated or reconfigured by a computer program stored in the computer. The procedures presented herein are not inherently related to a particular computer or other apparatus. Various general purpose machines may be used with programs written in accordance with the teachings herein, or it may prove more convenient to construct more specialized apparatus to perform the required method steps. The required structure for a variety of these machines will appear from the description given.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

**Figure 1A** illustrates a computer of a type suitable for carrying out the invention. Viewed externally in **Figure 1A**, a computer system has a central processing unit 100 having disk drives 110A and 110B. Disk drive indications 110A and 110B are merely symbolic of a number of disk drives which might be accommodated by the computer system. Typically, these would include a floppy disk drive such as 110A, a hard disk drive (not shown externally) and a CD ROM drive indicated by slot 110B. The number and type of drives varies, typically, with different computer configurations. The computer has a display 120 upon which information is displayed. A keyboard 130 and a mouse 140 are typically also available as input devices. Preferably, the computer illustrated in **Figure 1A** is a SPARC workstation from Sun Microsystems, Inc.

**Figure 1B** illustrates a block diagram of the internal hardware of the computer of **Figure 1A**. A bus 150 serves as the main information highway interconnecting the other components of the computer. CPU 155 is the central processing unit of the system, performing calculations and logic operations required to execute programs. Read only memory (160) and random access memory (165) constitute the main memory of the computer. Disk controller 170 interfaces one or more

disk drives to the system bus 150. These disk drives may be floppy disk drives, such as 173, internal or external hard drives, such as 172, or CD ROM or DVD (Digital Video Disks) drives such as 171. A display interface 175 interfaces a display 120 and permits information from the bus to be viewed on the display. Communications with external devices such as a network can occur over communications port 185.

Figure 1C illustrates an exemplary memory medium which can be used with drives such as 173 in Figure 1B or 110A in Figure 1A. Typically, memory media such as a floppy disk, CD ROM, or Digital Video Disk will contain the program information for controlling the computer to enable the computer to performs its functions in accordance with the invention.

Figure 2A is an illustration of one exemplary form of implementing the invention using a network such as an intranet. The network 200 is typically an internal organizational network that connects the client computing device and at least one WWW server.

Figure 2B is an illustration of another exemplary form of implementing the invention using a typical Internet arrangement. The client computing device 250 connects via one network with the client's Internet Service Provider (ISP) 240. The ISP 240 then connects

via the Internet 230 to a WWW server 260 requested by the user.

Figure 3 is an illustration of bandwidth allocation from a network server to several clients. The network server 300 has a predetermined amount of bandwidth N 310 which it must divide 320, 330, 340 and 350 between multiple clients 360, 370, 380, and 390. Note that the bandwidth allocation each client receives varies. This variance reflects the bandwidth allocation of the prioritized client connections as described in Figures 4-7, hereinafter.

Figure 4 is a database schema organized as an exemplary way for storing file-type priorities. The table has two columns: File Type 400 and Priority 410. An HTML file 420 will have a priority of 4 (430). A style sheet 440 will have a priority of 3 (450). Priority 2 470 is reserved for future use. GIF 480 and JPG files 490 both have priorities of 1 (485 and 495).

Figure 5 is a database schema organized as an exemplary way for storing connection information about clients actively involved in retrievals. The Allocation Utilization Table (AUT) 500 shown is a data structure used to track the status of each active current connection. For each such active current connection the file name 510, priority 520 (determined from Figure 4), allocated bandwidth 530, utilized bandwidth 540, and a

recalculation Boolean variable 550 are stored in the AUT. The AUT is used to provide the data for the bandwidth re-allocation algorithm of **Figure 7**. In rows 560 and 565 it should be noted that the allocated bandwidth exceeds the utilized bandwidth. When this difference exceeds a threshold, the recalculation variables are automatically set to FALSE to prevent re-allocation of more bandwidth than the connection can utilize.

**Figure 6** is a flowchart of a monitoring procedure to initiate dynamic bandwidth allocation by the server. A continuous monitoring loop 600 responds to events affecting bandwidth allocation. One event is a new request (605) for bandwidth. Normally, this will occur when an HTTP GET command is received by the server. Following a new request 605 the requested file name and its associated priority based on its file-type are placed into the AUT and the recalculation variable is set to TRUE (610). Then the recalculation of bandwidth allocation algorithm is invoked (650) which updates the AUT, then the AUT is used to provide parameters to the bandwidth allocator (655) and the monitoring loop resumes 600.

An event indicating the completion or cancellation of a transmission connection (615) will remove the connection from the AUT (620). Then the recalculation of bandwidth allocation algorithm is invoked (650) which

updates the AUT, the AUT is then used to provide parameters to the bandwidth allocator (655) and the monitoring loop resumes (600).

5 An event indicating that the client is not utilizing all of the allocated bandwidth occurs when the average throughput of data (calculated, for example, using the number of ACKs received per unit of time) falls below the allocated bandwidth maximum data rate (625). Various protocols from in the prior art, including Stop-And-Wait  
10 Link Utilization and Sliding-Window Flow Control, can be used to calculate the actual data rate. For example, one might add the packet lengths of a number of packets sent over a period of time and divide the total by the length of the period of time to determine effective throughput,  
15 or actual data rate. That value is then stored in the AUT and the recalculation variable is set to FALSE so that for the remainder of that retrieval request the bandwidth re-allocation algorithm will not increase that connection's bandwidth (630). Then the recalculation of  
20 bandwidth allocation algorithm is invoked (650) which updates the AUT, the AUT is then used to provide parameters to the bandwidth allocator (655) and the monitoring loop resumes (600).

25 An event indicating a change in the relative priority of a transmission (640) may occur. The priority for that transmission is then updated in the AUT (660).

Then the recalculation of bandwidth allocation algorithm is invoked (650) which updates the AUT, the AUT is then used to provide parameters to the bandwidth allocator (655) and the monitoring loop resumes (600).

5           A scheduled event may be set to occur periodically (645). This event is to handle any situation not handled by the other events. Then the recalculation of bandwidth allocation algorithm is invoked (650) which updates the AUT, the AUT is used to provide parameters to the bandwidth allocator (655) and the monitoring loop resumes (600).

10           **Figure 7A** is a flowchart of a procedure for dynamic bandwidth allocation by the server. The procedure begins by initializing several variables: MaxBW is set to the maximum bandwidth available to the server, COUNT is set to the number of rows in the AUT (i.e., the number of current connections), SumOfPriorities is set to zero, and index I (a loop counter) is set to one (700). If index I is not greater than COUNT (705) then there are more rows to process in the AUT. At 710 the MaxBW is decreased by any difference between the allocated bandwidth and the utilized bandwidth. If the AUT recalculation variable for the current row is TRUE then the priority of the current row is added to the SumOfPriorities (720). This action prevents those connections that are being under-utilized from receiving

more bandwidth which they have already demonstrated they cannot use. Then index I is incremented by one (725) and the loop continues at 705 until all the rows in the AUT have been processed.

5           Index I is re-initialized at 730 to one and another loop commences at 735. While index I does not exceed COUNT (735), the recalculation variable of each row is checked 760. If it is FALSE then the current AUT row has its allocated bandwidth set to equal its utilized bandwidth (755) thus reflecting the true state of the system. If it is TRUE, then the current AUT row allocated bandwidth variable is assigned the value of the ratio of the current row's priority to the SumOfPriorities and the utilized value is set to equal 10           the allocated value (765). This assumes that the client connection can utilize the new bandwidth it has been allocated. If it cannot, then it will be detected and corrected via Figure 6 at 625. In either case, index I is incremented (770) and processing the remaining rows in 15           the AUT continues at 735. If all the rows have been processed (735) then the AUT has been completely updated and is ready for use by the bandwidth allocator in Figure 6 at 655 and the process is terminated (750). 20

25           Figure 7B is a illustration depicting the result of one iteration of the dynamic bandwidth allocation procedure of Figure 7A. Assume a web server is



transmitting an HTML document and a JPG file to one client and a GIF file to another client. Using the bandwidth allocation algorithm described in **Figure 7A** and using the priority scheme of **Figure 4**, the sum of the priorities is 6 so the HTML document will receive 4/6 (66.7%) of the bandwidth, and both the JPG file and GIF file will receive 1/6 (16.7%) of the bandwidth. Now assume that the second client can only utilize 10% of the total bandwidth (which was determined using the process in **Figure 6** at 625). The initial state 780 is shown in rows 781, 782 and 783. Note that 783 has unused bandwidth capacity of 6.7% of the server's total bandwidth. In order for this bandwidth to be used, the bandwidth re-allocation algorithm of **Figure 7A** is run again.

The results of the reallocation are shown in the final state 790. Since the recalculation variable of the AUT row for the JPG file would have been set to FALSE, the sum of the priorities is now 5 and the proportional distribution of the spare bandwidth would be 4/5 for the HTML document 791 and 1/5 for the GIF document 792. So, 4/5 of the 6.7% available bandwidth is re-allocated to the HTML document, resulting in a final bandwidth utilization of 72.0%. Repeating the process for the GIF documents lead to a final bandwidth utilization of 18.0%.

The JPG file retains the 10.0% share it could use prior to the re-allocation.

**Figure 8A** is a database schema organized as an exemplary way for storing client browser-status priorities. Each browser has an ID **800**, a Status **810** and a Priority **815**. The row **820** has an ID of A, a status of "Has Focus" (i.e., is the user's active browser) and a high priority of 4. Browser ID B **821** does not have the focus but is visible on the screen and has a priority of 2. Browser ID C **823** does not have the focus and is not visible (perhaps minimized) and has a low priority of 1.

**Figure 8B** is a flowchart of a procedure for client-side control of bandwidth allocation. Similar to the algorithm of **Figure 7A**, this procedure utilizes the ratio of a process priority to that of the sum of the priorities of all active process. The MaxBW constant is initialized at **830** with the maximum bandwidth the client has available to manage. The priorities of the browsers in use are summed (**840**) and that sum is then used as the divisor of the individual browser priorities to determine the bandwidth to be allocated to each browser (**850**). Next, the bandwidth is allocated, e.g. by controlling the number of packets acknowledged to the server to obtain the desired throughput rate (**860**) (as described in **Figure 6** at **625**) and the process terminates (**870**).

**Figure 9** is a flowchart of a procedure for changing the priority of a connection after a fixed amount of data has been transmitted. As an alternative or a supplement to the preferred embodiment, an HTML file could be assigned a high priority only during the transmission of the first N KB. This approach might be used when only the first screen or part of the first screen of text must be delivered as rapidly as possible. The process begins by initializing N to the number of KB to transmit at high priority (900). Then the New Request event (910) (**Figure 6** at 605) is triggered. The number of KB transmitted is monitored (920) and when the number transmitted equals N (930) the AUT is updated to reflect a lower priority by triggering the Change Priority event (950) (**Figure 6** at 640).

**Figure 10A** is a database schema organized as an exemplary way for storing a value representing the variable amount of data a specific file needs have transmitted at high priority. As another alternative or supplement to the preferred embodiment, an HTML file could be examined (e.g., using a browser) to determine the number of bytes necessary to render the first page. This number of bytes would then be stored (1005) in a database along with the HTML file name 1000. The database rows 1006, 1007 and 1008 are examples of the name and byte tuples required.

**Figure 10B** is a flowchart of a procedure for changing the priority of a connection after the variable amount of data indicated in **Figure 10A** has been transmitted. The variable N is set, via a database lookup of the file name, to the number of bytes that need to be transmitted with high priority 1010. The New Request event is triggered (1020) (**Figure 6** at 605) and the number of bytes transmitted is monitored (1030). When the number of bytes transmitted equals N (1040) the Change Priority event is triggered (1060) (**Figure 6** at 640) which then updates the AUT table and begins the bandwidth re-allocation process and this process is terminated (1070).

**Figure 11A** is a database schema organized as an exemplary way for storing a list of customer passwords and a corresponding priority multiplier. As another supplement or alternative to the preferred embodiment, the priority given to web pages could be based on information about the person requesting them. For example, valued customers can be given higher priority. When implemented in a log-in type website, a list of passwords that have higher than normal priority could be kept in the form of the tuple Customer Password 1100 and Priority Multiplier 1105. As rows 1106, 1107 and 1108 show, the multiplier can be different based on how valued the customer is. In another approach, a list of network

addresses of valued customers is maintained and checked against the address of client's connections to the server to determine priorities.

**Figure 11B** is a flowchart of a procedure for increasing the priority of a connection using the information contained within **Figure 11A**. A customer accesses a website and enters his password (1110). If the password is in the priority database of **Figure 11A** (1120) then the priority for the document requested is set to the standard priority for that type of document multiplied by the priority multiplier 1130. For example, if an HTML document has a priority of 4 and the multiplier is 2 the new priority would be 8. Then the New Request event is triggered (1140) (**Figure 6** at 605) and this process is terminated (1140).

**Figure 12A** is a database schema organized as an exemplary way for storing a list of documents and their associated priority multipliers. As another supplement or alternative to the preferred embodiment, the priority given to web pages could be based on content of the pages themselves and their value to the web page owner. Providing an order form on the user's screen may be deemed to have a higher priority than delivering product information. The tuples of HTML page name 1200 and Priority Multiplier 1205 can be stored in the database. As rows 1206, 1207 and 1208 show, the multiplier can be

different based on how important a particular HTML document is.

**Figure 12B** is a flowchart of a procedure for increasing the priority of a connection using the information contained within **Figure 12A**. The user requests an HTML page (1210). If the page name is found in the priority database (1220) then the transmission priority becomes the normal transmission priority multiplied by the priority multiplier (1230). Then the New Request event is triggered (1240) (**Figure 6** at 605) and this process is terminated (1250).

There has thus been described a communication system in which communication resource allocated by either servers or clients can be adapted based on priority of various types. As a result, user satisfaction with the network is enhanced by obtaining desired information in a prompt fashion and server and client resources are prioritized to enhance throughput of the network.

Although the present invention has been described and illustrated in detail, it is clearly understood that the same is by way of illustration and example only and is not to be taken by way of limitation, the spirit and scope of the present invention being limited only by the terms of the appended claims.